Electronic Preprint for Journal of Information Processing Vol.27

Regular Paper

Melody2Vec: Distributed Representations of Melodic Phrases based on Melody Segmentation

Tatsunori Hirai[1,a)] Shun Sawada[2,b)]

Received: June 6, 2018, Accepted: December 4, 2018

Abstract: In this paper, we present *melody2vec*, an extension of the word2vec framework to melodies. To apply the word2vec framework to a melody, a definition of a word within a melody is required. We assume phrases within melodies to be words and acquire these words via melody segmentation applying rules for grouping musical notes called Grouping Preference Rules (GPR) in the Generative Theory of Tonal Music (GTTM). We employed a skip-gram representation to train our model using 10,853 melody tracks extracted from MIDI files primarily constructed from pop music. Experimental results show the effectiveness of our model in representing the semantic relatedness between melodic phrases. In addition, we propose a method to edit melodies by replacing melodic phrases within a musical piece based on the similarity of the phrase vectors. The naturalness of the resulting melody was evaluated via a user study and most participants who did not know the musical piece could not point out where the melody had been replaced.

Keywords: melody processing, distributed representations, word2vec, melody retrieval

1. Introduction

Melody is one of the most significant elements in music. Several methods for handling melodies using computers have been proposed. These methods can be roughly divided into two categories: symbolic processing approaches that directly handle melodies using notes and signal processing approaches that indirectly handle melodies using feature representations. While it is difficult to directly handle melodies with signal processing, the symbolic processing approach can focus on the melody itself. For example, the Generative Theory of Tonal Music (GTTM) [1] and the Implication-Realization Model (IRM) [2], [3] are well-studied methods in this field. In particular, a melody tree structure can be obtained by applying GTTM to handle melodies in a form corresponding to natural languages.

Natural language processing (NLP) is one of the most successful fields in symbolic processing. Even though there are multiple studies that apply NLP approaches to music information processing, the most successful method, called word2vec, cannot yet be applied to handle melodies well.

Word2vec is a model proposed by Mikolov et al. [4] that enables words to be represented by a vector. It converts symbolic words into numerical vectors. A melody is a semantically complex element; in fact, it used to be challenging to capture the intuitive and implicit similarity between melodies using conventional similarity measures [5]. It might be possible to visualize the semantic relationship between melodies with a melody vector. This might make it possible to quantitatively evaluate the quality of melodies. Because melodies are essential elements of music, vector representations of melodies are worth pursuing for future developments in music information retrieval (MIR) research fields.

However, the extension of the word2vec framework to melodies is not straightforward because the structures of a word and a melody are not the same. To extend the approach, we must first acquire a "word" within a melody. Because there is a correspondence between GTTM and natural languages, we can use it to acquire words in a melody. We assume melody phrases to be words and acquire them via melody segmentation using the Grouping Preference Rules (GPR) of GTTM.

In this paper, we propose a processing method to extend the word2vec framework to melodies including melody segmentation using GPR. In addition, we evaluate our model *melody2vec* and discuss possible applications.

2. Related Work

Multiple approaches to handle music in the same framework as NLP have been proposed. GTTM is a fundamental theory for describing the rules for melody generation, and many studies related to GTTM have been proposed [6], [7]. GTTM enables melodies to be symbolically analyzed by computer using a time-span tree. However, processing based on rules is not always successful because there are many exceptions in music. For example, conflicts between rules occur with GTTM and the priority depends on the context. A method that enables more flexible processing according to the data is therefore required.

Word2vec is successful NLP framework that learns the relationship between words and context from data [4]. Given large-scale data, the word2vec model learns word embedding for words having a similar meaning or ways of being used via methods

[1] Faculty of Global Media Studies, Komazawa University, Setagaya, Tokyo 154–8525, Japan
[2] Future University Hakodate, Hakodate, Hokkaido 041–8655, Japan
[a)] thirai@komazawa-u.ac.jp
[b)] g2116022@fun.ac.jp

called skip-gram and continuous bag-of-words (CBoW). These methods are powerful for learning semantic relationships. If we can handle melodies using a similar framework, related NLP methods developed by applying word2vec may be applicable to music information processing.

Several derived studies are applying the word2vec framework or a similar idea to other domains [8], [9], [10]. Illustration2vec [8] acquires vector representations of 2D illustrations via CNN. It enables estimations of annotations to be predicted, the retrieval of semantically similar illustrations, and a semantic morphing algorithm for two given illustrations. The word2vec framework has also been applied to the music domain. Music2vec [10] learns vectors of music by learning the context of listening to music pieces. The learned vector reflects a target user's preferences and is effective for music recommendation tasks.

Herremans et al. applied the word2vec framework to construct a semantic vector space model for capturing complex polyphonic musical contexts [11]. They extracted equal-length, non-overlapping slices including onset information from a polyphonic musical piece and treated them as words. In this method, the length of a slice is fixed and is not long enough to capture the essence of a melody. Therefore, the model can only consider the contextual information of short slices, which is not the direct modeling of a melody. A method to generate melodies using the LSTM model, which learns melodies assuming each note and its properties as a word has been proposed [12]. This method expresses the essence of a melody with a distributed representation. However, it assumes that a musical note is a word, which is too limited in its variations compared to word vocabularies in natural languages.

The biggest problem in applying the word2vec framework to melodies is how to define a "word" within a melody. Musical notes constituting a melody have elements such as a note name and a note value. It is possible to treat each of these elements as a letter in a word. In fact, there are description methods that express a melody as a sentence by converting musical elements into letters (symbols), e.g., Music Macro Language (MML). Assuming that a note is a letter and a melody is a sentence, what kind of musical concept does a word correspond to? Bernstein insisted that a word in natural language corresponds to a phrase of music [13]. In many sentences, important words repeatedly appear and can be considered as being close to phrases in a melody, which also repeatedly appear within a melody. Based on this property, in this paper, we assume phrases within a melody to be words and acquire them via melody segmentation.

Several methods for dividing melodies into phrases have been proposed [14]. These methods can be roughly divided into two categories: data-driven and model-driven approaches. A data-driven method determines the segmentation boundary based on the training data [15]. In a data-driven approach, the boundary of the phrase changes depending on the quality of the data. This is not preferable because the definition of the phrases changes with the training data. Model-driven methods include Lerdahl and Jackendoff's GPR in GTTM [1], Cambouropoulos's Local Boundary Detection Model (LBDM) [16], and Temperley's Grouper [17]. These methods are all related to the Gestalt principles. Of these methods, the effectiveness of GPR has been verified via psychological experiments [18]. By employing GPR in GTTM for melody segmentation, we can extend the word2vec framework to a melody.

3. Dataset

In a model using a neural network such as word2vec, the scale of the training data greatly affects the model accuracy. The number of melodies is preferably as large as possible. In the original word2vec paper by Mikolov et al. [4], the number of training data was 1.6 billion words. In this study, we use only music melodies as training data. To prepare as much training data as possible, we used the 178,561 songs in the Lakh MIDI dataset [19]. As far as the authors know, this dataset is the largest song dataset that can be used to analyze melodies. The Lakh dataset is constructed from MIDI files taken from the Internet.

If we can extract melodies from these MIDI data, we can easily prepare a large training data. Because the tracks are divided into each part of the music, the MIDI file does not require preprocessing, such as music source separation which is necessary to handle melodies within polyphonic audio data. However, according to Raffel et al. [20], there is no rule for specifying which of the tracks included in the MIDI file is the melody track. Therefore, we cannot explicitly extract the data of the melody tracks from all the 178,561 MIDI files, especially in the case of MIDI files "in the wild". It might be possible to use a program number that specifies the instrument for each track in a MIDI file to detect the melody track because many melody tracks are often played with certain instruments such as an acoustic piano or a synth voice. Even though it may be possible to extract some melody tracks using the program number, not all tracks played with those instruments are melody tracks. Therefore, using the program number for melody track identification will result in adding noise to the training data. Instead of using such an uncertain method, we chose a method that can extract melody tracks with high probability. We chose two keywords ("melody" and "vocal") and assumed that a track whose name included either of these two keywords as metadata was a melody track. As a result, we were able to extract melody tracks for 10,853 of the 178,561 songs in the dataset. We have evaluated the accuracy of melody extraction by checking for false positive results. Among the 10,853 extracted melody tracks, we randomly selected 250 tracks. We verified the accuracy of melody extraction, by manually listening to the original midi files of those 250 tracks and comparing with the extracted tracks. As a result, 238 out of 250 tracks were correctly extracted, which corresponds to 95.2% accuracy. Furthermore, 5 out of 12 incorrect tracks were empty tracks which did not affect our modeling process.

These 10,853 extracted melody tracks are all from the "wild" MIDI files taken from the Internet; therefore, the data contains large amounts of noise and fluctuations. Accordingly, the melody expression is ambiguous in terms of note duration. Even if the onset timing of a note in a melody track is correct, many data points have variations in the offset timing. For this reason, it is difficult to acquire a melody as per the musical score by simply recording the interval between the onset and the offset of the MIDI events.

Therefore, to acquire a melody as close as possible to the original musical score, in the case of a rest in which the time interval from offset to onset is shorter than an eighth note, we added the length of the short rest to the length of the previous note. In other words, the condition of the rest in our melody extraction process is that the silent section is an eighth rest or longer. Consequently, our analysis resulted in no rests of an unnatural length arising from the expression of the performance.

Using the 10,853 melodies extracted via the above procedure, we prepared the training data for melody2vec by applying preprocessing techniques including melody segmentation, key transposition, and octave normalization, which will be described in the next section.

4. Preprocessing

Phrases for learning the melody2vec model were acquired via the preprocessing techniques described in this section. Melodies extracted from a dataset were divided into phrases based on Gestalt by GPR in GTTM. We also transposed the keys in the melodies to either C major or A minor to normalize the meaning of each phrase.

4.1 Phrase Acquisition via Grouping Preference Rules in GTTM

The grouping rule in GTTM groups musical notes based on Gestalt cognition. Gestalt is a theory of perception where a certain meaning is found by integrating components that do not individually have meaning. In natural languages, an individual character does not have meaning but a word, a group of characters, has meaning. Similarly, in music, it is reasonable to define a melodic word by gathering musical notes based on Gestalt with the grouping rule in GTTM.

There are two grouping rules in GTTM, a grouping well-formedness rule (GWFR) and a GPR. GWFR must be satisfied to form a group. It prescribes that only a sequence of adjacent notes can form a group. GPR prescribes where the group boundaries exist. For example, a boundary tends to exist at a place where the length or dynamics of notes change. By applying this rule, melodies can be segmented into phrases.

There is a theory called exGTTM, which extends GTTM to be executable on a computer [21]. The exGTTM theory solves the ambiguity problem of the GTTM rules. We acquire the phrases of melodies referring to exGTTM.

According to GPR in exGTTM, when considering a sequence of four notes (i.e., n1, n2, n3, and n4), notes n_2 and n_3 are regarded as a boundary if the following conditions are satisfied:

$$ioi_{n2-n3} > ioi_{n1-n2} \quad \text{and} \quad ioi_{n2-n3} > ioi_{n3-n4} \tag{1}$$

where ioi_{n-m} represents an inter-onset-interval between the onsets of the n-th and m-th notes.

Following a preliminary experiment, we divided the melodies into phrases at the positions that satisfied these conditions. Since the note length is ambiguous due to the noise and fluctuations contained in the training data, this method is effective because it focuses only on the onset information of the notes. The result of applying the above segmentation method to a melody is shown in **Fig. 1**. In Fig. 1, we see that the melody is divided at a place where the notes change dynamically and that some phrases straddle a musical bar. If a melody is divided at every musical bar, a phrase that straddles a musical bar cannot be extracted.

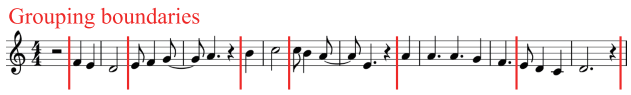

Fig. 1 Melody segmentation based on GPR in GTTM.

Table 1 The vocabulary size as a result of the preprocessing via each method.

key transposition method	vocabulary size	
	without octave normalization	with octave normalization
no transposition	318,783	306,192
Spiral Array Model	320,726	306,324
proposed transposition	302,502	**286,003**

4.2 Key Transposition and Octave Normalization

Even for phrases that are the same, their roles are different depending on the key of a musical piece. For example, a note sequence "G, A, B" appears in both C major and D major; however the role of "G, A, B" in D major is same as the role of "F, G, A" in C major. Therefore, by uniformly transposing the keys of musical pieces in training data to C major, the roles in the context originated by the key and the meaning of the phrases can be unified within the training data.

Similarly, for example, the difference between the note sequences "C4, D4, E4" and "C5, D5, E5" in the same key is only the octave. Although it depends on the context, the meaning of these note sequences is the same. Therefore, we normalize the octave to absorb the difference.

According to the survey of Raffel et al. [20], there are key tags in MIDI files but most of them are incorrect; therefore, we need to estimate the keys in order to transpose them. As a key transposition algorithm, we propose a histogram-based algorithm that transposes the key component of a melody to a direction that most matches the template of C major (C, D, E, F, G, A, B). Raffel et al. also mentioned that the keys included in the Lakh dataset are most likely to be major keys. Therefore, we assume that most of the keys are major keys and transposed them to C major. However, if the original key is a minor key, our algorithm transposes the key to A minor. Because A minor and C major both have the same key components (i.e., C, D, E, F, G, A, B), our algorithm transposes keys to either C major or A minor. Therefore, the roles of the phrases are still different between major and minor keys; however, the proportion of music with minor keys is small in the Lakh dataset.

The objective of key transposition is to maintain the functionality of the roles of phrases across the music dataset because a smaller phrase vocabulary is preferred. Although there are several methods to estimate keys from a melody [22], [23], [24], the vocabulary size acquired by our transposition method is smaller than that of other methods (**Table 1**). A smaller vocabulary also means that the variance of the phrase is small. Because the vocabulary of the phrases to be acquired from a melody tends to be larger than that of natural languages, a method to suppress the

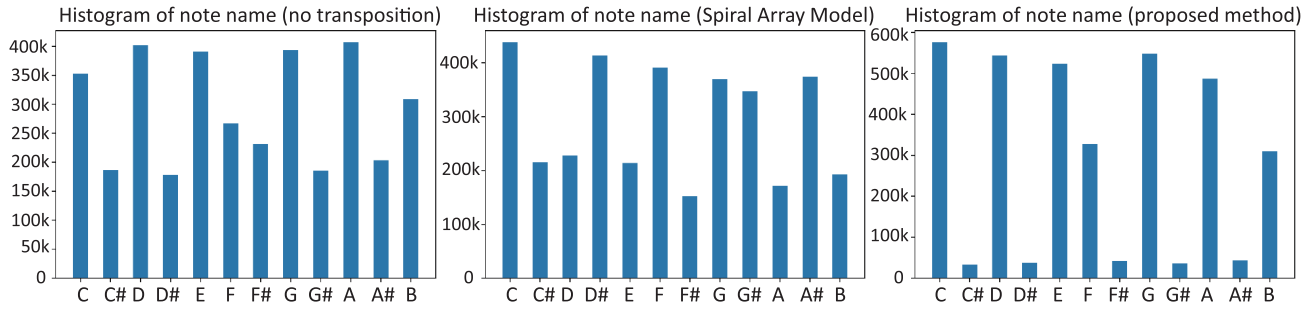

Fig. 2 Comparison of the note name histogram without key transposition (left), with key transposition using the Spiral Array Model (center), and with key transposition using the proposed method (right).

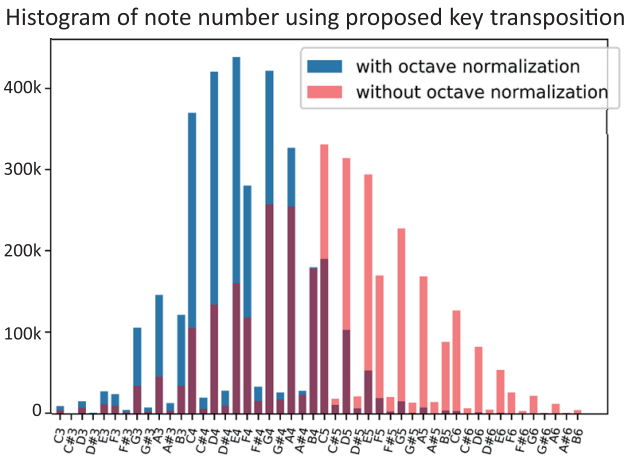

Fig. 3 Histogram of the note number before and after octave normalization.

vocabulary is desirable for effective model construction.

We compared our key transposition method to a method that we implemented referring to the Spiral Array Model (SAM) [24]. **Figure 2** shows histograms of musical notes included in the original melodies, melodies transposed by SAM, and the melodies transposed using our method. The histogram with our transposition method contains relatively more components of C major or A minor. Because the data contains noise, our simple transposition method performed better with respect to preprocessing for learning.

The SAM requires distinguishing between ♭ and ♯ symbols. We cannot distinguish these symbols from each other using the MIDI data and therefore, all ♭ and ♯ symbols were treated as ♯. This is one reason why we could not obtain high precision using SAM.

In addition to the key transposition, we normalized the octave of the notes. We calculated the most frequent octave within a piece and if it was the octave of "C3", "C5", or another octave, the melody was shifted to the "C4" octave. The histogram of the note number before and after the octave normalization is shown in **Fig. 3**. As shown in Fig. 3, the variance of the histogram is reduced by the octave normalization.

Table 1 shows the vocabulary size acquired using each method. The proposed transposition method with octave normalization has the smallest vocabulary, which corresponds to the smallest phrase variance. Therefore, we adopted histogram-based key transposition and octave normalization as our preprocessing technique.

Within the 286,003 phrases that we acquired by preprocessing, 151,006 phrases only occurred once in the dataset, which is 49.2% of the entire vocabulary. The ratio of phrases that occurred more than 5 times is 11.6%. As in word2vec modeling, our melody2vec modeling also requires phrases to occur frequently. Therefore, more than half the phrases cannot be sufficiently learned using our current dataset. To increase the number of phrases that can be successfully learned, we need to improve the number of melody tracks which can be used for the modeling.

5. Melody2Vec

The melody2vec model was constructed using the 957,628 melodic phrases acquired by the preprocessing as training data. The vocabulary size of the training data was 286,003. The melody2vec model adopts a skip-gram model, as proposed by Mikolov et al. [4] in their original word2vec model. Because the data structure of a melodic phrase is completely the same as a word in a sentence, we can simply apply the training algorithm of the word2vec model. Following preliminary experiments using different parameters, we set the vector dimension to 100 and the context window size to 3.

After training, we tested the quality of the melody vectors by searching for the nearest neighbors of an arbitrary query phrase. **Table 2** shows the result of the top three similar phrases to an arbitrary query phrase. The similarity here is the cosine similarity between two melody vectors. The expression "C4 : 1/4" represents the quarter note of the C4 note. As shown in Table 2, some phrases do not have a similar component but the similarity is high. In addition, the similarities of phrases having similar components are high, which means that the similarities of similar phrases are high. Surprisingly, phrases no.1 and 2 for the query phrase 1 only occurred twice in the dataset. Phrase no.1 for the query phrase 2 also occurred only twice and phrase no.2 occurred only once in the dataset. This indicates that in some circumstances the frequency of phrase appearance is enough with two or less.

We validate the effect of this similarity in terms of the semantic relatedness of phrases in the next section.

6. Results

6.1 Visualization

6.1.1 Visualization via 2D Mapping

Since there is no ground truth data on the similarity of melody phrases, it is difficult to accurately measure the effectiveness of

Table 2 Top three phrases with similarity to a query phrase.

query phrase 1	D4:1/16 - G4:1/16 - G4:1/16 - D4:1/16 - G4:1/8	similarity
No. 1	D4:1/16 - G4:1/8 - D4:1/16 - R:4/4	0.711
No. 2	A4:1/16 - B4:1/8 - G4:1/8 - R:1/8	0.692
No. 3	D4:1/16 - G4:1/8	0.655
query phrase 2	G3:1/8 - C4:1/8 - D4:1/8 - C4:1/8 - D4:1/4	similarity
No. 1	G3:1/8 - C4:1/8 - D4:1/8 - C4:1/8 - D4:1/6 - R:4/4	0.858
No. 2	D4:1/8 - D4:1/8 - G4:1/6 - F4:1/6 - D4:1/4	0.761
No. 3	D4:1/8 - G3:1/8	0.679
query phrase 3	C4:1/6 - R:3/8 - E4:1/8 - D4:1/8 - D4:1/8 - D4:1/8 - C4:1/6 - R:1/4	similarity
No. 1	C4:1/6 - R:3/8 - E4:1/8 - D4:1/8 - D4:1/8 - D4:1/8 - C4:1/16 - A3:1/16 - C4:1/12 - R:1/4	0.999
No. 2	C4:1/8 - D4:1/8 - E4:1/4 - E4:1/4 - E4:1/8 - D4:1/4 - R:3/8	0.752
No. 3	E4:1/8 - D4:1/8 - D4:1/8 - D4:1/8 - C4:1/6 - R:1/4	0.745

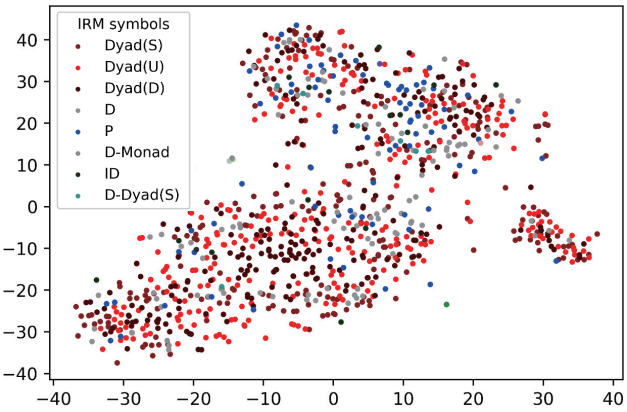

Fig. 4 2D embedding visualization via t-SNE.

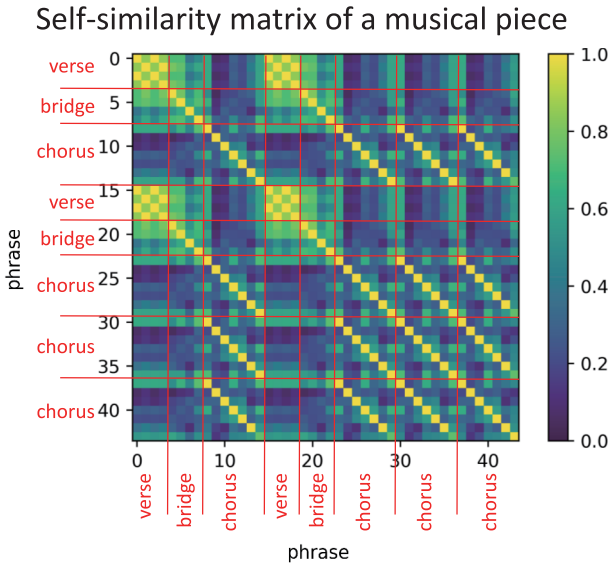

Fig. 5 Self-similarity matrix of a musical piece.

melody vector. In this section, we evaluated our result using 2D mapping visualization. We visualized a set of melody phrases by projecting the 100-dimensional vector space onto a 2D space using t-SNE [25]. **Figure 4** shows a plot of 1,000 frequently occurring phrases in the training data. Since melody2vec modeling does not work well with phrases that occur with low frequency in a dataset, we only visualized frequently occurring phrases to avoid distractions caused by noisy plots. To maintain visibility in the figure, we did not label every single plot. However, additional information is essential to explain the effect of visualization. Instead of a phrase label, we colored each plot based on the symbols given by IRM [2], [3]. Originally 10 types of symbols were proposed in IRM. These symbols are assigned to a group of three consecutive notes. Therefore, more than one IRM symbol can be assigned to phrases having more than three notes. We describe the melody phrase by combinations of IRM symbols. To the 1,000 frequently occurring phrases, 19 types of IRM symbol label patterns were assigned. Because there are many types of symbol patterns, only the top eight frequently appearing symbol patterns are displayed in the legend in Fig. 4. Since IRM symbols describe characteristics of melodies, phrases with the same IRM symbol patterns are similar.

Basically, IRM symbols will be assigned to three consecutive musical notes within a melody phrase. Conversely, among the 1,000 phrases, 71.1% of phrases are phrases composed of two musical notes. In the original IRM paper, a symbol called a "dyad" was assigned to any two sequential notes. There is no discrimination within dyad; however, there are many dyad phrases in our sample, so we distinguished these phrases from others by way of the interval between the two notes. Dyad (S) indicates that two notes are the same, with the second note being higher in a dyad (U) and lower in a dyad (L). Other symbols are the same as those in the original IRM. We distinguished differences within dyad because we did not want to treat symbols of two notes similarly just because they are two note phrases.

Although the plot is not clustered for each color, it can be seen that adjacent plots are often of the same color. The color of the plots indicates that similar melodies are placed nearby. The closest plot in 37.6% of the 1,000 plots was the same symbol. Plots with the same symbols in the neighboring three plots accounted for 68.8% of plots. This number is significant because there are 19 types of IRM symbols in the top 1,000 phrases.

6.1.2 Visualization of the Similarity within a Song

We examined how the similarity based on the melody vector changes within a song. After acquiring the melody vectors from a song in the training data, the cosine similarity between the melody vectors within the song was calculated. We examined the song "Good Riddance" by Green Day. **Figure 5** shows the self-similarity matrix. The composition of the song is {verse → bridge → chorus → verse → bridge → chorus (3 times)}. The yellow part indicates phrases with high similarity and reflects the structures of the song. Notably, the verse and bridge of this song are composed of different but similar phrases.

6.2 User Study

We performed a user study to verify the effectiveness of our

Table 3 The proposed similarity and Levenshtein distances of the phrase set used in the user study.

	Proposed similarity of		Levenshtein distance of	
No.	phrase A	phrase B	phrase A	phrase B
1	0.903	2.160×10^{-6}	10	4
2	0.917	2.270×10^{-6}	12	9
3	0.999	8.703×10^{-2}	5	4
4	0.911	1.740×10^{-6}	10	7
5	0.901	1.080×10^{-7}	6	5
6	0.901	2.040×10^{-6}	9	8
7	0.904	9.570×10^{-7}	8	6
8	0.916	4.430×10^{-7}	12	6
9	0.985	1.860×10^{-8}	10	7
10	0.901	1.080×10^{-7}	6	5
avg.	0.924	8.704×10^{-3}	8.8	6.1

melody vector. We verified the effectiveness indirectly by evaluating the similarity between vectors.

A total of 17 participants were recruited for the user study. In each trial, we let the participants listen to three types of melodies, namely Q, A, and B. The phrase Q is a query, A is the nearest phrase to Q according to the cosine similarity between the proposed vectors (hereafter, referred to as the proposed similarity), and B is the furthest phrase from Q according to the proposed similarity but is closer than A according to the Levenshtein distance (also referred to as the edit distance). Participants were asked to select the melody most similar to Q from A or B for 10 sets. The A to B listening order was shuffled for every trial to minimize effects of listening in sequence.

Table 3 shows the proposed similarity and Levenshtein distance of all the Q-A-B sets. 10 random queries were chosen but phrases that occurred less than 10 times in the training data were ignored. We also discarded queries when the nearest vector's proposed similarity was less than 0.9. This was to ensure that Q and A were similar enough in terms of proposed similarity because, in some query phrases, sufficiently similar phrases do not exist. As shown in Table 3, the proposed similarity of A was higher than 0.9. In addition, if the Levenshtein distance between the query and the furthest phrase was larger than that of A, we searched for the second furthest and so on until a phrase with a smaller Levenshtein distance was found. As shown in Table 3, the Levenshtein distance of B is smaller than that of A. B is a phrase that is not similar in terms of the proposed similarity but is more similar in terms of the Levenshtein distance. In the calculation of the Levenshtein distance, we handled the note name and the note value separately. If a note in a phrase changes from "C4:1/4" to "C4:1/8", the Levenshtein distance is 1, and if it changes to "D4:1/8", the Levenshtein distance is 2. Here, we did not set the weight to the insertion or deletion cost in calculation of the Levenshtein distance.

The results of the user study are shown in **Table 4**. The number in the table is the percentage of participants who selected the phrase (the sum of A and B is 100%). Most participants selected the phrase according to the proposed similarity (A) being similar to the query rather than that of the Levenshtein distance (B) (89.4% of the selected answers were phrases chosen by the proposed similarity).

Phrases with small Levenshtein distances are similar because they have multiple common notes. This result indicates that our similarity measure is effective in finding similar phrases and that, if the similarity is low, it is a phrase that is not similar even if the Levenshtein distance is small, i.e., the phrase with multiple points in common. It may be useful to find a phrase that is different but similar using the proposed method.

Table 4 The result of the user study verifying the effectiveness of the proposed similarity measure.

	Similarity measure	
No.	Proposed (A)	Levenshtein (B)
1	100.0	0.0
2	94.1	5.9
3	88.2	11.8
4	76.5	23.5
5	100.0	0.0
6	88.2	11.8
7	70.6	29.4
8	100.0	0.0
9	88.2	11.8
10	88.2	11.8

Because ground truth data for the similarity of melodies does not yet exist, further verification of the semantic relatedness between melodies will be done in future studies.

7. Melody Replacement Using Melody2Vec

As an application of the melody2vec model, we introduce the replacement of phrases in a melody. As shown in Table 2 in Section 5, it is possible to search for another phrase that is similar to the query phrase using the proposed similarity. An arbitrary melody phrase of an existing musical piece can be replaced with another phrase using this similar phrase retrieval.

Here, the phrase in a musical piece have a fixed length; therefore we add constraints on the length of a phrase when searching for a phrase to replace the query phrase. Therefore, the candidate for replacement is the one with the highest similarity between phrases having the same length as the query phrase. **Figure 6** shows an example of melody replacement. The piece is "All my loving" by the Beatles, which was included in the training data. We chose the first phrase "F4:1/4 - E4:1/4 - D4:1/2" in the melody as the phrase to be replaced. The phrase was replaced with "D4:1/4 - E4:1/8 - C4:3/8 - R:1/4". which is totally different phrase. Here, the proposed similarity between the phrases was 0.82.

We conducted a user study to verify the naturalness of the replaced melody. The 21 participants (13 males, 8 females) were asked to listen to the four melodies, namely melody A, B, C and D. These melodies are melodies in which a phrase has been replaced. Before listening to them, the participants were told that the melody had been modified. We asked the participants to note the playback time when the replacement was performed after listening to the melody. The answer was given in units of seconds. If the answer was within 1 second of the replacement section, we regarded it as a correct answer. When determining their answer, we let the participants listen to the melody as many times as they wanted.

Melody A was the melody in Fig. 6 including the continuous part. The length of the melody A was approximately 24 seconds. Melody B was a part of melody from "Let it be" by the Beatles which was included in the training data; its length was approxi-

Table 5 Evaluation of melody replacement.

	Overall accuracy	Overall evaluation of naturalness	Accuracy rate by prior knowledge		Accuracy rate by naturalness	
			People who knew the melody	People who did not know the melody	People who answered the melody natural	People who answered the melody unnatural
Melody A	0.24	0.43	0.60	0.13	0.22	0.25
Melody B	0.38	0.81	0.62	0.00	0.41	0.25
Melody C	0.48	0.81	0.53	0.00	0.35	1.00
Melody D	0.62	0.67	0.65	0.00	0.57	0.71
Total	0.43	0.68	0.60	0.07	0.40	0.48

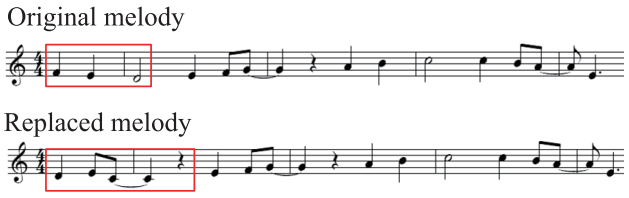

Fig. 6 Musical score before and after the melody replacement.

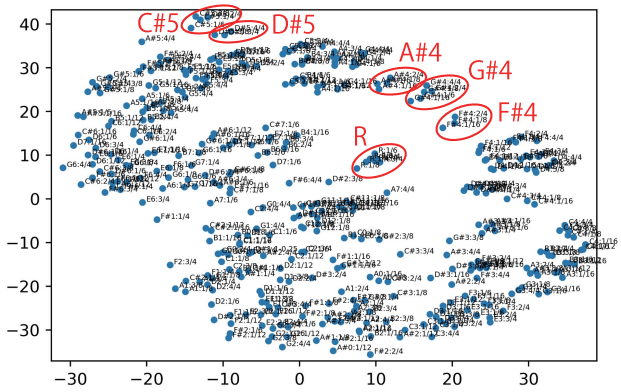

Fig. 7 Visualization of 1-note vector using t-SNE (400 randomly selected plots).

mately 25 seconds. Melodies C and D were melodies that were not included in the training data, and the songs are "March 9th" by Remioromen and "Koi" by Gen Hoshino and their lengths were approximately 37 and 27 seconds, respectively. Note that we selected these melodies on the basis that the replacement result is not similar to the original phrase. As can be seen from Table 2, the similarity tends to be high when a phrase is literally similar. However, the purpose of melody replacement is to change the phrase to totally different phrase. Therefore, we chose four clear results from several tested musical pieces.

In addition to the question about pointing out the replaced phrase, we asked the participants if they knew or had heard the melodies before. We also evaluated the naturalness of melody replacement by asking the participants if melodies are natural.

The result of the user study is shown in **Table 5**. The accuracy rate for people who knew the song was higher than that of people who did not know the song. Further, most of the people who did not know the song could not answer correctly (only two participants answered correctly for melody A). As the evaluation of naturalness, 68% of answers suggest that melodies are natural. In most cases, people who answered the melody was unnatural could answer more correctly than people who answered the melody was natural. However, people who answered the melody natural often answered incorrectly. As can be assumed from this result, there is a possibility that participants may have felt the melody unnatural for parts other than the replaced section. This may be caused by the incompleteness of melody extracted from MIDI files in the wild. From these results, we can say that our method has the potential to naturally modify a melody to a different one.

In addition to the evaluation experiment, we have asked a professional pianist to review our melody replacement results (melody A, B, C, and D). According to the comments, the pianist did not find the unnatural part in the replaced melodies because the chord progression was natural. She has mentioned that prerequisite that there is an original melody may have been effective to feel melodies natural.

In future studies, we plan to create an interface where a user can specify a phrase to be replaced and be presented with candidate phrases for the replacement. By specifying the phrases that a user wants to change within an existing song and displaying the candidate replacement phrases, we will let a user who does not have experience in composing music to interactively compose a new melody.

8. Note2vec

In the previous chapters, we have described melody2vec. In this section we introduce other modeling method as a variation. Here we redefined a word in melody by regarding one note as a word and construct note2vec model.

From the 10,853 extracted melody tracks, we acquired 3,992,279 one-note words. The vocabulary has 691 words. The preprocessing, i.e., key transposition and octave normalization, were done in a same way as explained in Section 4.2. The modeling process is done as same to the one described in the Section 5.

As a note2vec modeling result, we have visualized vectors via 2D mapping using t-SNE. **Figure 7** is a visualization of randomly selected 400 phrases (i.e., notes). In this figure, we labeled every single plot. As seen from the figure, there are clusters of the same note number. Also clusters of note in the same octave tends be close. From this result, we can see that the note modeling also works well with word2vec framework.

9. Conclusions

We present *melody2vec*, a method to extend the word2vec framework to melodies. Our melody segmentation using GPR in GTTM makes it possible to apply the word2vec framework to melodies. We constructed a melody2vec model with over 10,000 melody tracks and verified its effectiveness via a user study. In addition, we present an example of a melody2vec application that enabled natural melody modifications.

9.1 Limitations

In our current preprocessing technique, we cannot distinguish

between major and minor keys. To further improve the effectiveness of the model, it is necessary to estimate the key more accurately. Currently, we are only using 6% of the MIDI files in the Lakh dataset. To receive further benefits of such large-scale data, it might prove effective to use melody track identification methods such as the method proposed by Rizo et al. [26].

Currently, our model is for a phrase in a melody, which is not the melody itself. Other post-processing techniques are required to acquire a distributed representation of the entire melody. For example, it may be effective to construct a vector using a doc2vec approach [27].

Since melody part is mainly included in a song, the scope of application of our method is limited to the melody parts of musical pieces i.e., mostly the vocal part. Therefore, our method is not applicable to contemporary music and classical music at the moment. It might be possible to extend our method to other instrumental parts by applying the method to other instrumental parts. To achieve the same performance in other instruments, it is required to extract the track data of same instrument played in same performance style.

9.2 Discussion

In Mikolov et al.'s original word2vec paper [4], they presented a vector arithmetic of words (e.g., "king" - "man" + "woman" ≈ "queen"). We have not been able to verify the validity of the melody vector arithmetic. Although we have implemented such operations, we were not able to verify if the result was reasonable, because we could not imagine what the calculation results for a melody would be like.

Because many songs have lyrics that consist of a collection of words, it is possible to acquire both a melody vector and a word vector from a song. Such a distributed representation helps handle different information in the same manner. This approach might improve the accuracy of the song retrieval.

A distributed representation for a melody phrase can be an essential factor in MIR and has tremendous potential for future music information processing research not only in retrieval but also in all other applications using melodies. Therefore, we will further explore the possibilities of the melody vector.

Acknowledgments We would like to thank Sakurako Yazawa for her advice on IRM symbol labeling and Yuko Kujime for her review of melody replacement result.

References

[1] Lerdahl, F. and Jackendoff, R.S.: *A Generative Theory of Tonal Music*, The MIT press (1983).
[2] Narmour, E.: *The Analysis and Cognition of Basic Melodic Structures*, The University of Chicago Press (1990).
[3] Narmour, E.: *The Analysis and Cognition of Melodic Complexity*, The University of Chicago Press (1992).
[4] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S. and Dean, J.: Distributed Representations of Words and Phrases and Their Compositionality, *Advances in neural information processing systems*, pp.3111–3119 (2013).
[5] Volk, A., Van Kranenburg, P., Garbers, J., Wiering, F., Veltkamp, R.C. and Grijp, L.P.: A Manual Annotation Method for Melodic Similarity and the Study of Melody Feature Sets, *Proc. International Symposium on Music Information Retrieval*, pp.101–106 (2008).
[6] Hamanaka, M., Hirata, K. and Tojo, S.: Melody Morphing Method Based on GTTM, *Proc. International Computer Music Conference*, pp.155–158 (2008).
[7] Hirata, K. and Matsuda, S.: Interactive Music Summarization based on Generative Theory of Tonal Music, *Journal of New Music Research*, Vol.32, No.2, pp.165–177 (2003).
[8] Saito, M. and Matsui, Y.: Illustration2Vec: A Semantic Vector Representation of Illustrations, *SIGGRAPH Asia 2015 Technical Briefs*, *SA '15*, pp.5:1–5:4, ACM (online), DOI: 10.1145/2820903.2820907 (2015).
[9] Vosoughi, S., Vijayaraghavan, P. and Roy, D.: Tweet2Vec: Learning Tweet Embeddings Using Character-level CNN-LSTM Encoder-Decoder, *Proc. 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp.1041–1044 (2016).
[10] Wang, D., Deng, S. and Xu, G.: Sequence-based Context-aware Music Recommendation, *Information Retrieval Journal*, pp.1–23 (2017).
[11] Herremans, D. and Chuan, C.-H.: Modeling Musical Context with Word2vec, *Proc. 1st International Conference on Deep Learning and Music*, pp.11–18 (2017).
[12] Shin, A., Crestel, L., Kato, H., Saito, K., Ohnishi, K., Yamaguchi, M., Nakawaki, M., Ushiku, Y. and Harada, T.: Melody Generation for Pop Music via Word Representation of Musical Properties, arXiv preprint arXiv:1710.11549 (2017).
[13] Bernstein, L.: Lecture II, Musical Syntax, in "Unanswered Question" (1973).
[14] López, M.R. and Volk, A.: Automatic Segmentation of Symbolic Music Encodings: A Survey (2012).
[15] Bod, R.: Memory-based Models of Melodic Analysis: Challenging the Gestalt Principles, *Journal of New Music Research*, Vol.31, No.1, pp.27–36 (2002).
[16] Cambouropoulos, E.: The Local Boundary Detection Model (LBDM) and its Application in the Study of Expressive Timing, *ICMC* (2001).
[17] Temperley, D.: *The Cognition of basic Musical Structures*, MIT press (2004).
[18] Deliege, I.: Grouping Conditions in Listening to Music: An Approach to Lerdahl & Jackendoff's Grouping Rreference Rules, *Music Perception: An Interdisciplinary Journal*, Vol.4, No.4, pp.325–359 (1987).
[19] Raffel, C.: *Learning-based Methods for Comparing Sequences, with Applications to Audio-to-midi Alignment and Matching*, PhD Thesis, Columbia University (2016).
[20] Raffel, C. and Ellis, D.P.: Extracting Ground-Truth Information from MIDI Files: A MIDIfesto, *Proc. International Symposium on Music Information Retrieval*, pp.796–802 (2016).
[21] Hamanaka, M., Hirata, K. and Tojo, S.: ATTA: Implementing GTTM on a Computer, *Proc. International Symposium on Music Information Retrieval*, pp.285–286 (2007).
[22] Longuet-Higgins, H.C. and Steedman, M.J.: On Interpreting Bach, *Machine intelligence*, Vol.6, pp.221–241 (1971).
[23] Castellano, M.A., Bharucha, J.J. and Krumhansl, C.L.: Tonal Hierarchies in the Music of North India, *Journal of Experimental Psychology: General*, Vol.113, No.3, pp.394–412 (1984).
[24] Chew, E.: *Towards a mathematical model of tonality*, PhD Thesis, Massachusetts Institute of Technology (2000).
[25] van der Maaten, L. and Hinton, G.: Visualizing Data using t-SNE, *Journal of Machine Learning Research*, Vol.9, No.Nov, pp.2579–2605 (2008).
[26] Rizo, D., Leon, P., Perez-Sancho, C., Pertusa, A. and Inesta, J.: A Pattern Recognition Approach for Melody Track Selection in MIDI Files, *Proc. International Symposium on Music Information Retrieval*, pp.61–66 (2006).
[27] Le, Q. and Mikolov, T.: Distributed Representations of Sentences and Documents, *Proc. International Conference on Machine Learning*, pp.1188–1196 (2014).

Tatsunori Hirai was born in 1988. He received his B.S., M.S. and Ph.D. degrees from Waseda University in 2011, 2012, and 2015, respectively. He joined the Information Processing Society of Japan in 2010. He is currently an assistant professor at faculty of global media studies, Komazawa University, Tokyo, Japan. His research interest is multimedia content processing.

Shun Sawada was born in 1993. He received his B.S. and M.S. degrees from Future University Hakodate in 2016 and 2017, respectively. He joined the Information Processing Society of Japan in 2014. He is currently a doctoral course student at Future University Hakodate, Hokkaido, Japan. His research interest is music information processing.